Package 'neat'

February 1, 2024

Title Efficient Network Enrichment Analysis Test

Version 1.2.4

Description Includes functions and examples to compute NEAT, the Network Enrichment Analysis Test described in Signorelli et al. (2016, <DOI:10.1186/s12859-016-1203-6>).

License GPL-3

URL https://mirkosignorelli.github.io/r

Depends R (>= 4.1.0)

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.2.3

Imports igraph

Suggests knitr, Matrix, rmarkdown

NeedsCompilation no

Author Mirko Signorelli [aut, cre, cph] (<https://orcid.org/0000-0002-8102-3356>), Veronica Vinciotti [ctb], Ernst Wit [ctb]

Maintainer Mirko Signorelli <msignorelli.rpackages@gmail.com>

Repository CRAN

Date/Publication 2024-02-01 09:30:02 UTC

R topics documented:

`neat-package` *neat*

Description

Includes functions and examples to compute NEAT, the Network Enrichment Analysis Test described in Signorelli et al. (2016).

Author(s)

Mirko Signorelli

References

Signorelli, M., Vinciotti, V., Wit, E. C. (2016). NEAT: an efficient network enrichment analysis test. BMC Bioinformatics, 17:352. Url: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-016-1203-6.

See Also

`neat`

`neat` *Performs neat for lists of gene sets*

Description

Compute NEAT (Signorelli et al., 2016), a test for network enrichment analysis between/from a first list of sets ('A sets') and/to a second list of sets ('B sets').

Usage

```
neat(alist, blist = NULL, network, nettype, nodes,
alpha = NULL, mtc.type = 'fdr', anames = NULL, bnames = NULL)
```

Arguments

`alist`	List of A sets. Each element within the list is a vector of genes and represents a gene set
`blist`	List of B sets. Each element within the list is a vector of genes and represents a gene set. If `nettype == "undirected"`, this argument is optional: if provided, every set of `blist` is compared with every set of `alist`; if `NULL`, the function compares sets in `alist` between themselves

network	One of the following objects: an adjacency matrix of class "matrix" (see 'Example 1') or a sparse adjacency matrix of class "dgCMatrix"; an igraph object (see 'Example 2'); a two-column matrix where every row represents and edge (for directed networks, parent nodes must be in the first column, and child nodes in the second)
nettype	Either 'directed' or 'undirected'
nodes	Vector containing the (ordered) names of all nodes in the network
alpha	Significance level of the test (optional). If specified, a column with the conclusion of the test is added to the output
mtc.type	Type of multiple testing correction (NB: added from package version 1.2.0). Use 'fdr' or 'BH' for the Benjamini-Hockberg method, and 'none' if no multiple testing correction is required. To know the shortcuts for other multiple testing correction methods, see p.adjust
anames	Vector of names for the elements of alist (optional: it has to be provided only if the elements of alist are not named)
bnames	Vector of names for the elements of blist (optional: it has to be provided only if the elements of blist are not named)

Value

A data frame with the following columns:

A	A set
B	B set
nab	observed number of links from A to B
expected_nab	expected number of links from A to B (in absence of enrichment)
pvalue	p-value of the test
adjusted.p	p-value adjusted to account for multiple testing
conclusion	conclusion of the test (only if alpha is specified): no enrichment, overenrichment or underenrichment

Author(s)

Mirko Signorelli

References

Signorelli, M., Vinciotti, V., Wit, E. C. (2016). NEAT: an efficient network enrichment analysis test. BMC Bioinformatics, 17:352. Url: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-016-1203-6.

See Also

networkmatrix, plot.neat, print.neat, summary.neat

Examples

```
# Example 1: network given as adjacency matrix:
A = matrix(0, nrow=7, ncol=7)
A[1,c(2,3)]=1; A[2,c(5,7)]=1;A[3,c(1,4)]=1;A[4,c(2,5,7)]=1;A[6,c(2,5)]=1;A[7,4]=1
labels = letters[1:7]
set1 = c('a','e')
set2 = c('c','g')
set3 = c('d','f')
alist = list('set 1' = set1, 'set 2' = set2)
blist = list('set 3' = set3)

# test without multiple testing correction
test1 = neat(alist = alist, blist = blist, network=A,
             nettype='directed', nodes=labels,
             alpha=0.05, mtc.type = 'none')
print(test1)

# test with FDR multiple testing correction (default)
test1 = neat(alist = alist, blist = blist, network=A,
             nettype='directed', nodes=labels,
             alpha=0.05, mtc.type = 'fdr')
print(test1)

# Example 2: network given as igraph object:
library(igraph)
network = erdos.renyi.game(15, 1/3)
set1 = 1:4
set2 = c(2,5,13)
set3 = c(3,9,14)
set4 = c(8,15,20)
alist = list('set 1' = set1, 'set 2' = set2)
blist = list('set 3' = set3, 'set 4' = set4)

test2 = neat(alist, blist, network = network,
             nettype='undirected', nodes=seq(1,15),
             alpha=NULL)
print(test2)

# Example 3: network given as list of links:
networklist = matrix(nrow=13, ncol=2)
networklist[,1]=c('a','a','b','b','c','d','d','d','f','f','f','h','h')
networklist[,2]=c('d','e','e','g','d','b','e','g','a','b','e','c','g')

labels = letters[1:8]
set1 = c('a','b','e')
set2 = c('c','g')
set3 = c('d','f')
set4 = c('a','b','f')
alist = list('set 1' = set1, 'set 2' = set2)
blist = list('set 3' = set3, 'set4' = set4)

test3 = neat(alist, blist, network = networklist,
```

```
              nettype = 'undirected', nodes=labels,
              alpha=0.05, mtc.type = 'none')
print(test3)

alist = list('set 1' = set1, 'set 2' = set2, 'set 3' = set3)
test4 = neat(alist, network = networklist,
              nettype = 'undirected', nodes=labels,
              alpha=0.05, mtc.type = 'none')
print(test4)

# Example 4: ESR data
## Not run:
data(yeast)
esr = list('ESR 1' = yeast$esr1, 'ESR 2' = yeast$esr2)
test = neat(alist = esr, blist = yeast$goslimproc, network = yeast$yeastnet,
            nettype = 'undirected', nodes = yeast$ynetgenes, alpha = 0.01)
# Replace with "blist = yeast$kegg" to use kegg pathways

m = dim(test)[1]
test1 = test[1:(m/2),]
table(test1$conclusion)
plot(test1)
o1=test1[test1$conclusion=='Overenrichment',]
print(o1, nrows='ALL') #display overenrichments

test2 = test[(m/2+1):m,]
table(test2$conclusion)
plot(test2)
o2=test2[test2$conclusion=='Overenrichment',]
print(o2, nrows='ALL') #display overenrichments

## End(Not run)
```

`networkmatrix` *Creates a network matrix for* `neat`

Description

Internal function, creates a two-column network matrix that can be further processed by `neat`.

Usage

```
networkmatrix(network, nodes, nettype)
```

Arguments

`network`	One of the following objects: an adjacency matrix (class `"matrix"`), a sparse adjacency matrix (class `"dgCMatrix"`) or an igraph graph (class `"igraph"`)
`nodes`	Vector containing the (ordered) names of all nodes in the network
`nettype`	Either `'directed'` or `'undirected'`

Details

This is an internal function, that is called within neat to convert different types of network objects (see argument 'network' above) into a standard two-column network matrix, that can then be processed by neat.

Value

A two-column matrix, where every row represents and edge. For directed networks, parent nodes must be in the first column, and child nodes in the second.

Author(s)

Mirko Signorelli

References

Signorelli, M., Vinciotti, V., Wit, E. C. (2016). NEAT: an efficient network enrichment analysis test. BMC Bioinformatics, 17:352. Url: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-016-1203-6.

See Also

neat

Examples

```
# First case: adjacency matrix
n<-50
adjacency <- matrix(sample(0:1, n^2, replace=TRUE, prob=c(0.9,0.1)), ncol=n)
diag(adjacency) <- 0
lab = paste(rep('gene'),1:n)
head(networkmatrix(adjacency, lab, 'directed'))

# Second case: sparse adjacency matrix
library(Matrix)
sparse_adjacency<-Matrix(adjacency,sparse=TRUE)
head(networkmatrix(sparse_adjacency, lab, 'directed'))

# Third case: igraph object
library(igraph)
igraph_graph = erdos.renyi.game(15, 1/3)
lab = paste(rep('gene'),1:15)
head(networkmatrix(igraph_graph, lab, 'directed'))
```

plot.neat *Plot method of* neat

Description

plot method for class "neat".

Usage

```
## S3 method for class 'neat'
plot(x, nbreaks = 10, ...)
```

Arguments

x	An object of class "neat"
nbreaks	Number of breaks to be used in the histogram (default is 10)
...	Further arguments passed to or from other methods

Value

An histogram showing the distribution of p-values and a p-p plot comparing the distribution of p-values to the uniform distribution.

Author(s)

Mirko Signorelli

References

Signorelli, M., Vinciotti, V., Wit, E. C. (2016). NEAT: an efficient network enrichment analysis test. BMC Bioinformatics, 17:352. Url: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-016-1203-6.

See Also

neat, print.neat, summary.neat

Examples

```
## Not run:
data(yeast)
esr2 = list('ESR 2' = yeast$esr2)

test = neat(alist = esr2, blist = yeast$goslimproc, network = yeast$yeastnet,
            nettype='undirected', nodes = yeast$ynetgenes, alpha = 0.01)

plot(test)

## End(Not run)
```

print.neat *Print method of* neat

Description

print method for class "neat".

Usage

```
## S3 method for class 'neat'
print(x, nrows=10, ...)
```

Arguments

x	An object of class "neat"
nrows	Maximum number of results to print (default is 10). It can be either an integer number or "ALL"
...	Further arguments passed to or from other methods

Value

A dataframe showing the first nrows tests contained in a neat object.

Author(s)

Mirko Signorelli

References

Signorelli, M., Vinciotti, V., Wit, E. C. (2016). NEAT: an efficient network enrichment analysis test. BMC Bioinformatics, 17:352. Url: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-016-1203-6.

See Also

neat, plot.neat, summary.neat

Examples

```
A = matrix(0, nrow=7, ncol=7)
A[1,c(2,3)]=1; A[2,c(5,7)]=1;A[3,c(1,4)]=1;A[4,c(2,5,7)]=1;A[6,c(2,5)]=1;A[7,4]=1

labels = letters[1:7]
set1 = c('a','e')
set2 = c('c','g')
set3 = c('d','f')
alist = list('set 1' = set1, 'set 2' = set2)
blist = list('set 3' = set3)
```

```
test = neat(alist, blist, network=A, nettype='directed', nodes=labels, alpha=0.05)
print(test)
```

---

summary.neat *Summary method of* neat

Description

summary method for class "neat".

Usage

```
## S3 method for class 'neat'
summary(object, ...)
```

Arguments

object	An object of class "neat"
...	Further arguments passed to or from other methods

Value

The summary.neat function returns the following values:

- the number of tests computed;
- the number of enrichments at 1% and 5% level;
- the p-value of the Kolmogorov-Smirnov test to check if the distribution of p-values is uniform.

Author(s)

Mirko Signorelli

References

Signorelli, M., Vinciotti, V., Wit, E. C. (2016). NEAT: an efficient network enrichment analysis test. BMC Bioinformatics, 17:352. Url: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-016-1203-6.

See Also

neat, plot.neat, summary.neat

Examples

```
## Not run:
data(yeast)
esr = list('ESR 1' = yeast$esr1, 'ESR 2' = yeast$esr2)
test = neat(alist = esr, blist = yeast$goslimproc, network = yeast$yeastnet,
            nettype = 'undirected', nodes = yeast$ynetgenes, alpha = 0.01)

test1 = test[1:99,]
summary(test1)

test2 = test[100:198,]
summary(test2)

## End(Not run)
```

---

yeast *List collecting various yeast data (see 'description')*

Description

`yeast` is a list that contains:

`yeastnet`: network matrix representing Yeastnet-v3 (Kim et al., 2013)

`ynetgenes`: vector with the names of the genes appearing in `yeastnet`

`esr1`: vector containing the first of the two gene sets that constitute the "Environmental Stress Response" (ESR) reported by Gasch et al. (2012)

`esr2`: vector containing the second gene set of the ESR

`goslimproc`: list containing the gene sets of the GOslim process ontology (Ashburner et al., 2000) for the buddying yeast Saccaromyces Cerevisiae (groups 'biological process' and 'other' are not included)

`kegg`: list containing the KEGG pathways (Kanehisa and Goto, 2002) for the buddying yeast Saccaromyces Cerevisiae

Format

`yeast`: list

Source

Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., Davis, A. P., Dolinski, K., Dwight, S. S., Eppig, J. T., et al. (2000). Gene ontology: tool for the unification of biology. Nat. Genet., 25(1), 25-29.

Gasch, A. P., Spellman, P. T., Kao, C. M., Carmel-Harel, O., Eisen, M. B., Storz, G., Botstein, D., and Brown, P. O. (2000). Genomic expression programs in the response of yeast cells to environmental changes. Mol. Biol. Cell, 11(12), 4241-4257.

Kanehisa, M., and Goto, S. (2002). KEGG: Kyoto Encyclopedia of Genes and Genomes. Nucleic Acids Res., 28(1), 27-30.

Kim, H., Shin, J., Kim, E., Kim, H., Hwang, S., Shim, J. E., and Lee, I. (2013). Yeastnet v3: a public database of data-specific and integrated functional gene networks for saccharomyces cerevisiae. Nucleic Acids Res., 42 (D1), D731-6.

Signorelli, M., Vinciotti, V., Wit, E. C. (2016). NEAT: an efficient network enrichment analysis test. BMC Bioinformatics, 17:352. Url: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-016-1203-6.

References

Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., Davis, A. P., Dolinski, K., Dwight, S. S., Eppig, J. T., et al. (2000). Gene ontology: tool for the unification of biology. Nat. Genet., 25(1), 25-29.

Gasch, A. P., Spellman, P. T., Kao, C. M., Carmel-Harel, O., Eisen, M. B., Storz, G., Botstein, D., and Brown, P. O. (2000). Genomic expression programs in the response of yeast cells to environmental changes. Mol. Biol. Cell, 11(12), 4241-4257.

Kanehisa, M., and Goto, S. (2002). KEGG: Kyoto Encyclopedia of Genes and Genomes. Nucleic Acids Res., 28(1), 27-30.

Kim, H., Shin, J., Kim, E., Kim, H., Hwang, S., Shim, J. E., and Lee, I. (2013). Yeastnet v3: a public database of data-specific and integrated functional gene networks for saccharomyces cerevisiae. Nucleic Acids Res., 42 (D1), D731-6.

Signorelli, M., Vinciotti, V., Wit, E. C. (2016). NEAT: an efficient network enrichment analysis test. BMC Bioinformatics, 17:352. Url: https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-016-1203-6.

See Also

neat

Examples

```
## Not run:
data(yeast)
esr = list('ESR 1' = yeast$esr1, 'ESR 2' = yeast$esr2)
test = neat(alist = esr, blist = yeast$goslimproc, network = yeast$yeastnet,
            nettype = 'undirected', nodes = yeast$ynetgenes, alpha = 0.01)
# Replace with "blist = yeast$kegg" to use kegg pathways

m = dim(test)[1]
test1 = test[1:(m/2),]
o1=test1[test1$conclusion=='Overenrichment',]
# list of overenrichments for the first ESR set:
print(o1, nrows='ALL')

test2 = test[(m/2+1):m,]
o2=test2[test2$conclusion=='Overenrichment',]
# list of overenrichments for the second ESR set:
print(o2, nrows='ALL')
```

```
# the same can be done using KEGG pathways:
keggtest = neat(alist = esr, blist = yeast$kegg, network = yeast$yeastnet,
            nettype = 'undirected', nodes = yeast$ynetgenes, alpha = 0.01)

## End(Not run)
```

Index